This listing of claims will replace all prior versions, and listings, of claims in the application.

Listing of Claims:

1. (Currently Amended) A method for providing an interface description for a service of a device or object in a computing system, wherein the method is implemented by at least one processor of the computing system, comprising:

creating a one to one mapping of each abstract type in the device or object to an XML schema type, said mapping comprising:

a one to one association between the abstract type to said XML schema type;

a one to one association between said XML schema type to said abstract type,

whereby there is a one to one relationship between ~~an instance of the abstract type~~ the service of the device or object to an XML document said one to one relationship being characterized in a way that the service of the device or object is an instance of the abstract type if and only if the XML document is valid in accordance with the XML schema ~~so that an Is Instance operator between said abstract type and an instance returns TRUE if and only if an Is Valid operator between the corresponding XML schema type and XML Document returns TRUE~~;

describing the one to one mapping with an extensible markup language (XML)-based Type Description Language having a grammar for representing behavioral aspects of said abstract type and said XML schema type.

2. (Previously Presented) A method according to claim 1, wherein said Type Description Language accommodates classes that have data and behavioral aspects.

3. (Previously Presented) A method according to claim 2, wherein said element of creating a one to one mapping comprises creating a one to one mapping from a programming construct to an XML schema for describing the programming construct.

4. (Original) A method according to claim 3, wherein the programming construct is one of a pointer programming construct, primitive type programming construct, struct programming construct, class programming construct, array programming construct, subtype programming construct, enumeration type programming construct, service reference construct and bit field programming construct.

5. (Previously Presented) A method according to claim 2, wherein said element of creating a one to one mapping comprises creating a one to one mapping from a constant value of complex type to an XML schema for describing the constant value of complex type and defining a constant value global attribute in said Type Description Language.

6. (Previously Presented) A method according to claim 2, wherein said element of creating a one to one mapping comprises creating a one to one mapping of actions, services, interfaces, methods, properties and event sources from the abstract type to the XML schema type.

7. (Previously Presented) A method according to claim 3, wherein Type Description Language supports inheritance of programming constructs.

8. (Previously Presented) A method according to claim 1, wherein the Type Description Language is a wire format for message communications relating to the service between devices of the computing system.

9. (Previously Presented) A method according to claim 8, further comprising creating a one to one mapping from the wire format to the message communications.

10. (Previously Presented) A method according to claim 2, wherein Type Description Language enables a transfer of a service reference across an application boundary.

11. (Original) A method according to claim 1, wherein the computing system is a peer to peer distributed computing environment.

12. (Previously Presented) A method according to claim 1, wherein the XML-based Type Description Language has action elements, service elements, interface elements, method elements, property elements and event source elements.

13. (Currently Amended) A tangible computer readable medium having stored thereon a plurality of computer-executable instructions for performing the method of claim 1.

14. (Canceled).

15. (Original) A computing device comprising means for performing the method of claim 1.

16. (Currently Amended) A tangibly embodied computer readable medium having stored thereon a plurality of computer-executable modules, the computer executable modules including at least one mechanism implemented by at least one processor of a computing system, the at least one mechanism comprising:

a mapping mechanism for describing a service of one of a device and object in a computing system with an extensible markup language (XML)-based Interface Description Language (IDL) that one to one maps each type of a particular type-based system to an XML schema so that there is a one to one mapping from the abstract type of said type-based system to said XML schema type and vice-versa and a one to one mapping from said service of one of a device and object ~~an instance of the abstract type~~ to an XML document so that said service of one of a device and object is an instance of the abstract type if and only if the XML document is valid in accordance with the XML schema ~~the Is Instance operator between said abstract type and an instance returns TRUE if and only if the Is Valid operator between the corresponding XML Schema Type and XML Document returns TRUE~~.

17. (Previously Presented) A computer readable medium according to claim 16, wherein the XML-based Interface Description Language is a Type Description Language having a grammar for representing behavioral aspects of said abstract type and said XML schema type.

18. (Previously Presented) A computer readable medium according to claim 17, wherein Type Description Language enables a one to one mapping from a programming construct to an XML schema for describing the programming construct.

19. (Original) A computer readable medium according to claim 18, wherein the programming construct is one of a pointer programming construct, primitive type programming construct, struct programming construct, class programming construct, array programming construct, subtype programming construct, enumeration type programming construct, service reference construct and bit field programming construct.

20. (Previously Presented) A computer readable medium according to claim 17, wherein said Type Description Language enables a one to one mapping from a constant value of complex type to an XML schema for describing the constant value of complex type and vice versa.

21. (Previously Presented) A computer readable medium according to claim 17, wherein said Type Description Language enables a one to one mapping from at least one of properties, methods and events of the type system to an XML schema for describing the at least one of properties, methods and events and vice versa.

22. (Previously Presented) A computer readable medium according to claim 18, wherein said Type Description Language supports inheritance of programming constructs.

23. (Previously Presented) A computer readable medium according to claim 16, wherein the Type Description Language is a wire format of message communications relating to the service between devices of the computing system.

24. (Previously Presented) A computer readable medium according to claim 23, wherein the Type Description Language enables a one to one mapping from the wire format to the message communications and vice versa.

25. (Previously Presented) A computer readable medium according to claim 17, wherein said Type Description Language enables a transfer of a service reference across an application boundary.

26. (Original) A computer readable medium according to claim 16, wherein the computing system is a peer to peer distributed computing environment.

27. (Previously Presented) A computer readable medium according to claim 16, wherein the mapping mechanism for the Type Description Language has action elements, service elements, interface elements, method elements, property elements and event source elements.

28. (Canceled).

29. (Original) A computing device comprising means for carrying out the plurality of computer-executable instructions of the computer readable medium of claim 16.

30. (Currently Amended) A computing device, comprising:

computer-executable instructions tangibly embodied on a computer readable medium, the computer-executable instructions of the operating system including at least one mechanism implemented by at least one processor of a computing system, the at least one mechanism comprising:

a mapping mechanism for describing a service of one of a device and object in a computing system with an extensible markup language (XML)-based Interface Description Language that maps each abstract type of a particular type-based system to an XML schema so that there is a one to one mapping from the abstract type of said type-based system to a type in said XML schema and vice-versa and a one to one mapping from said service of one of a device and object ~~an instance of the abstract type~~ to an XML document so that said service of one of a device and object is an instance of the abstract type if and only if the XML document is valid in accordance with the XML schema ~~the Is Instance operator between said abstract type and an instance returns TRUE if and only if the Is Valid operator between the corresponding XML Schema Type and XML Document returns TRUE~~.

31. (Previously Presented) A computing device according to claim 30, wherein the XML-based Interface Description Language is Type Description Language.

32. (Previously Presented) A computing device according to claim 31, wherein Type Description Language enables a one to one mapping from a programming construct to an XML schema for describing the programming construct.

33. (Original) A computing device according to claim 32, wherein the programming construct is one of a pointer programming construct, primitive type programming construct, struct programming construct, class programming construct, array programming construct, subtype programming construct, enumeration type programming construct, service reference construct and bit field programming construct.

34. (Previously Presented) A computing device according to claim 31, wherein said Type Description Language enables a one to one mapping from a constant value of complex type to an XML schema for describing the constant value of complex type.

35. (Previously Presented) A computing device according to claim 31, wherein said Type Description Language enables a one to one mapping from at least one of properties, methods and events of the type system to an XML schema for describing the at least one of properties, methods and events.

36. (Previously Presented) A computing device according to claim 32, wherein said Type Description Language supports inheritance of programming constructs.

37. (Previously Presented) A computing device according to claim 30, wherein the XML-based Interface Description Language is a wire format of message communications relating to the service between devices of the computing system.

38. (Previously Presented) A computing device according to claim 37, wherein the XML-based Interface Description Language enables a one to one mapping from the wire format to the message communications.

39. (Previously Presented) A computing device according to claim 31, wherein Type Description Language enables a transfer of a service reference across an application boundary.

40. (Original) A computing device according to claim 30, wherein the computing system is a peer to peer distributed computing environment.

41. (Previously Presented) A computing device according to claim 30, wherein the mapping mechanism for the XML-based Interface Description Language has action elements, service elements, interface elements, method elements, property elements and event source elements.